# Advanced Computer Systems Project 3 Report

Ivan Cheng (ECSE)

Rensselaer Polytechnic Institute

ECSE 4320 Advanced Computer Systems

Dr. Tong. Zhang

October 1, 2025

# Setup/methodology

**Test Environment:**

The experiments were performed on an Intel Core i7-11850H laptop. The laptop runs on Tiger Lake-H, 8 cores/16 threads, 2.5 Ghz base.

Memory: 32 GB Ram DDR4
Storage: Samsung MZVL2512HCJQ-00BL7 - 512GB Pcie TLC M.2 2280 SSD
OS: Windows
Benchmark tool: Microsoft DiskSpd

A 16 GiB file (testfile.dat) was created to ensure workloads were large enough to pass windows file caching and engage the SSD.

Benchmark:

1. Zero-queue baselines
   Measure QD=1 latency for: (a) 4 KiB random reads & writes, (b) 128 KiB sequential reads & writes.
2. Block-size sweep (pattern fixed)
   Sweep 4 KiB→256 KiB random and sequential runs.
   Produce IOPS/MB/s and average latency
3. Read/Write mix
   100%R, 100%W, 70/30, 50/50 ratios
4. Queue-depth sweep
   QD values: 1, 2, 4, 8, 16, 32, 64
   Single throughput vs. latency trade-off curve. Identify the knee & relation to Little's Law
5. Tail Latency
   DiskSpd's percentile reporting (-Rxml) was enabled to capture p50/p95/p99/p99.9 latencies.

PSL commands:
-Sh disables OS cache, -L enables latency statistics

```
.\diskspd.exe -c1G -d10 -b4k   -r -t1 -o1 -w100 -Sh -L -Rtext C:\Temp\4k_rand_write.txt
.\diskspd.exe -c1G -d10 -b128k -s -t1 -o1 -w0   -Sh -L -Rtext C:\Temp\128k_seq_read.txt
.\diskspd.exe -c1G -d10 -b128k -s -t1 -o1 -w100 -Sh -L -Rtext C:\Temp\128k_seq_write.txt
.\diskspd.exe -c1G -d10 -b4k   -r -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\rand_4k.txt
.\diskspd.exe -c1G -d10 -b16k  -r -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\rand_16k.txt
.\diskspd.exe -c1G -d10 -b32k  -r -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\rand_32k.txt
.\diskspd.exe -c1G -d10 -b64k  -r -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\rand_64k.txt
.\diskspd.exe -c1G -d10 -b128k -r -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\rand_128k.txt
.\diskspd.exe -c1G -d10 -b256k -r -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\rand_256k.txt
.\diskspd.exe -c1G -d10 -b4k   -s -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\seq_4k.txt
```

```
.\diskspd.exe -c1G -d10 -b16k  -s -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\seq_16k.txt
.\diskspd.exe -c1G -d10 -b32k  -s -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\seq_32k.txt
.\diskspd.exe -c1G -d10 -b64k  -s -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\seq_64k.txt
.\diskspd.exe -c1G -d10 -b128k -s -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\seq_128k.txt
.\diskspd.exe -c1G -d10 -b256k -s -t1 -o4 -w0 -Sh -L -Rtext C:\Temp\seq_256k.txt
.\diskspd.exe -c1G -d10 -b4k -r -t1 -o4 -w0   -Sh -L -Rtext C:\Temp\mix_4k_100R.txt
.\diskspd.exe -c1G -d10 -b4k -r -t1 -o4 -w100 -Sh -L -Rtext C:\Temp\mix_4k_100W.txt
.\diskspd.exe -c1G -d10 -b4k -r -t1 -o4 -w30  -Sh -L -Rtext C:\Temp\mix_4k_70R30W.txt
.\diskspd.exe -c1G -d10 -b4k -r -t1 -o4 -w50  -Sh -L -Rtext C:\Temp\mix_4k_50R50W.txt
.\diskspd.exe -c1G -d10 -b4k -r -t1 -o1  -w0 -Sh -L -Rtext C:\Temp\qd1_read.txt
.\diskspd.exe -c1G -d10 -b4k -r -t1 -o2  -w0 -Sh -L -Rtext C:\Temp\qd2_read.txt
.\diskspd.exe -c1G -d10 -b4k -r -t1 -o4  -w0 -Sh -L -Rtext C:\Temp\qd4_read.txt
.\diskspd.exe -c1G -d10 -b4k -r -t1 -o8  -w0 -Sh -L -Rtext C:\Temp\qd8_read.txt
.\diskspd.exe -c1G -d10 -b4k -r -t1 -o16 -w0 -Sh -L -Rtext C:\Temp\qd16_read.txt
```

# Zero-queue Baselines

Accurate average & percentile latencies for 4 KiB random and 128 KiB sequential (R/W)

| Test Case | IOPS (k) | Bandwidth (MB/s) | Avg Latency (ms) | P50 Latency (ms) | p95 Latency (ms) | p99 Latency (ms) | P99.9 Latency (ms) |
|---|---|---|---|---|---|---|---|
| 4 KiB random read, QD=1 | 27,595 | 107.8 | .036 | .031 | .038 | .051 | .139 |
| 4 KiB random write, QD=1 | 27,545 | 107.6 | .036 | .032 | .039 | .053 | 0.117 |
| 128 KiB sequential read, QD=1 | 14,416 | 1802 | .069 | .059 | .071 | .096 | 0.283 |
| 128 KiB sequential write, QD=1 | 11,960 | 1495 | .083 | .062 | .080 | .118 | .644 |

Based on my data in my zero-queue baseline experiments, data is expected as when comparing read vs writes reads are faster at larger blocks. In the random 4KiB experiments, data shows low latency and limited bandwidth. At higher blocks (128KiB) experiment, there is higher bandwidth with higher latency. At QD = 1 we can see there is no queuing delay.
p50 ≈ pure device service time
Tail (p99/99.9) represents infrequent device/OS events.
In terms of SLA implications, p99.9<= .1ms will miss as I measured .139ms in my 4 KiB write.
P99 <= .06ms for 4KiB reads will most likely be fine as I measured .051ms.
For 128KiB reads my p99.9 was .283ms so reads < 200ms would miss.
For the 128KiB write I noticed that the p99.9 jumped to .644 this spike may have been due to background OS scheduling, side delays or other factors.

# Block-size Sweep

| Block Size | Access Pattern | IOPS | Bandwidth (MB/s) | Avg Latency (ms) |
|---|---|---|---|---|
| 4KiB | Random | 66,554 | 260 | .036 |
| 4KiB | Seq | 65527 | 256 | .027 |
| 16KiB | Random | 65231 | 1019 | .041 |
| 16KiB | Seq | 64133 | 1002 | .041 |
| 32KiB | Random | 66329 | 2073 | .035 |
| 32KiB | Seq | 66310 | 2072 | .035 |
| 64KiB | Random | 61144 | 3822 | .049 |
| 64KiB | Seq | 43993 | 2750 | .085 |
| 128KiB | Random | 49904 | 6238 | ..077 |
| 128KiB | Seq | 36540 | 4567 | .105 |
| 256KiB | Random | 24667 | 6167 | .161 |
| 256KiB | Seq | 22250 | 5563 | .174 |

Based on my data, numbers match expected storage scaling behavior. Small blocks have high IOPS with low throughput MB/s and low latency. When compared to medium blocks they have lower IOPS with higher throughput in MB/s. In the larger block cases IOPS drops even further and MB/s saturates as the SSD hits the maximum throughput it can deliver. Latency increases as operations require more transfer of data. When comparing random and sequential access patterns, the sequential data usually hits higher throughput at higher block sizes since data is written in read/write order and the ssd can use internal prefetching and coalescing. In smaller blocks random and sequential processes perform similarly.

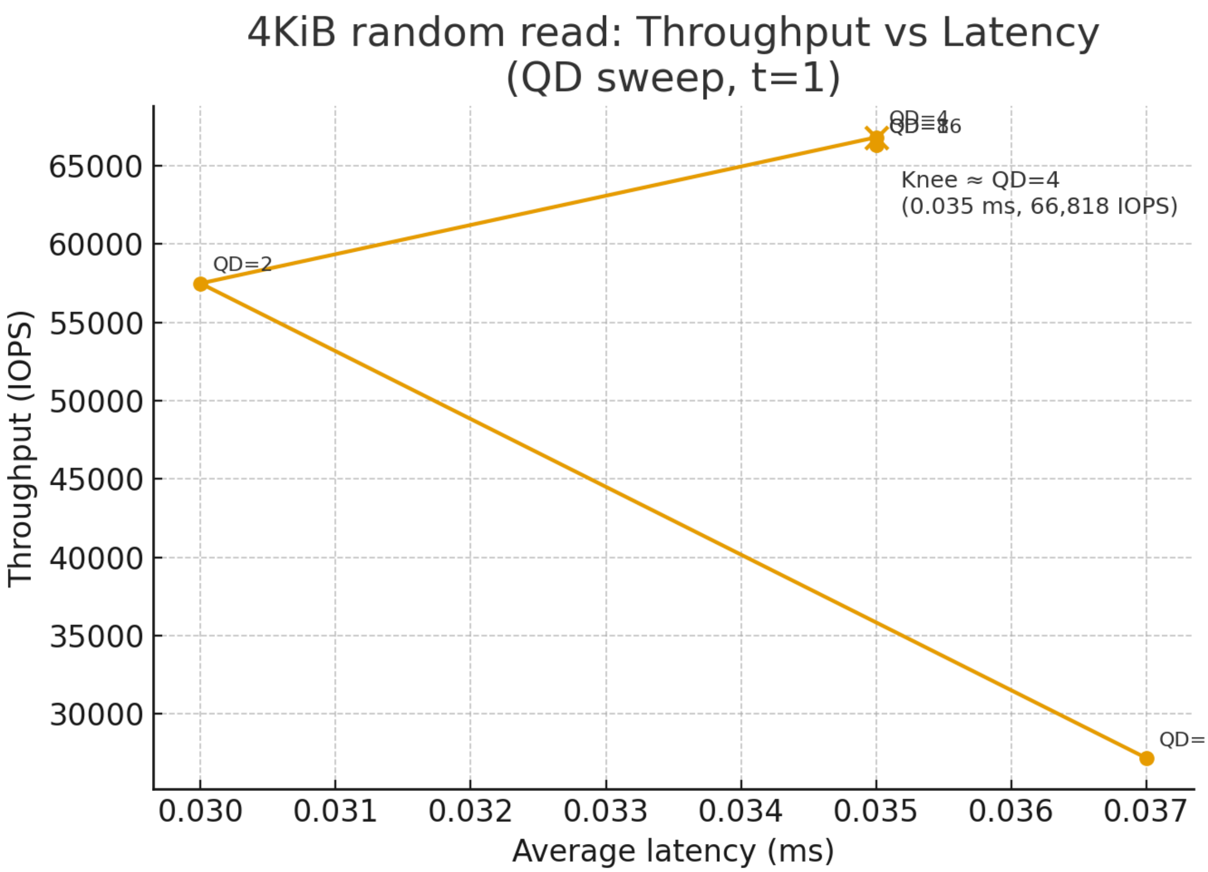
# Read/write mix

| Mix Ratio | IOPS (k) | Bandwidth (MB/s) | Avg Latency (ms) | P95 Latency (ms) | P99 Latency (ms) |
|---|---|---|---|---|---|
| 100% Read | 66613 | 260.2 | .036 | .051 | .057 |
| 100% Write | 59592 | 232.8 | .032 | .040 | .052 |
| 70R/30W | 50027 | 195.4 | .068 | .098 | .967 |
| 50R/50W | 63127 | 246.6 | .046 | .089 | .126 |

Based on the data received through running my experiments, Pure read/write cases deliver the highest throughput and lowest latencies. In the mixed workload cases both runs have reduced IOPS and higher tail latencies because writes contend with reads and flash must erase in addition to reading. My 70/30 case is worse than my 50/50 case in terms of tail latency most likely due to controller scheduling prioritizing writes differently.

# Queue-depth sweep

| QD | IOPS (k) | Bandwidth (MB/s) | Avg Latency (ms) | P95 Latency (ms) | P99 Latency (ms) |
|---|---|---|---|---|---|
| QD = 1 | 27173 | 106.14 | .037 | .037 | .047 |
| QD = 2 | 57490 | 224.57 | .030 | .040 | .055 |
| QD = 4 | 66818 | 261.01 | .035 | .051 | .055 |
| QD = 8 | 66313 | 259.04 | .035 | .051 | .056 |
| QD = 16 | 66327 | 259.09 | .035 | .051 | .056 |




With my data, it is observed that going from QD 1 to QD 2 over doubles the IOPS. At QD 4 it goes up to 66k IOPS and afterwards starts to saturate and QD 8 and 16 show no improvement. QD 1 has an average latency of .037 ms and improves to .03 at QD = 2. At QD >=4 average latency stabilizes to .035ms with P95 and P99 latency around .05ms & .056ms respectively. The SSD saturates around QD = 4 which means additional parallelism does not yield more throughput.

In relation to Little's Law Throughput ≈ Concurrency/Latency it is predicted that when QD goes past 4 throughput would increase. However measured results stay around 66k. This divergence indicates the device limit for this workload has been reached and additional I/O sits queued. Effective service rate caps out at QD = 4 and throughput no longer follows Little Law.

# Anomalies/Limitations

During the course of this project a few anomalies and limitations were encountered:

1. WSL environment issues.
   Initially, the experiments were conducted under WSL. However there were issues with getting results with installing FIO onto WSL and issues with how WSL handles system calls and disks I/O. As a result testing was moved to DiskSpd on native Windows.
2. PSL write execution results
   When attempting to automate multiple tests at once in one file using PSL scripts an anomaly occurred where write results would output 0 IOPS, 0 MB/s essentially all outputs of write were showing 0. To resolve this each test command was ran manually in Powershell instead of relying on the automated loop.

# Conclusion

This project evaluated storage performance under a range of benchmarks using Microsoft's DiskSpd. The results highlighted differences between random vs. sequential access, block-size scaling, read/write mixes, and queue depth behavior. Overall, while DiskSpd provided reproducible measurements, there were anomalies throughout this project such as automation in PSL and environment setup in WSL.

# References

[1] OpenAI, "ChatGPT," ChatGPT, 2025. https://chatgpt.com/
[2]microsoft, "Releases · microsoft/diskspd," GitHub, Jun. 13, 2024. https://github.com/microsoft/diskspd/releases
[3] "Intel® CoreTM i7-11850H Processor (24M Cache, up to 4.80 GHz)," Intel, 2025. https://www.intel.com/content/www/us/en/products/sku/213799/intel-core-i711850h-processor-24m-cache-up-to-4-80-ghz/specifications.html?wapkw=i7-11850H